class ClassA {

private int aValue;

public ClassA(int aValue) {

this.aValue = aValue;

System.out.println("ClassA constructor called: aValue = " + aValue);

}

public ClassA(ClassA other) {

this.aValue = other.aValue;

}

public int getAValue() {

return aValue;

}

public void setAValue(int aValue) {

this.aValue = aValue;

}

}

class ClassB extends ClassA {

protected int bValue;

protected ClassB(int aValue, int bValue) {

super(aValue);

this.bValue = bValue;

System.out.println("ClassB constructor called: bValue = " + bValue);

}

public ClassB(ClassB other) {

super(other);

this.bValue = other.bValue;

}

}

class ClassC extends ClassB {

public int cValue;

public ClassC(int aValue, int bValue, int cValue) {

super(aValue, bValue);

this.cValue = cValue;

System.out.println("ClassC constructor called: cValue = " + cValue);

}

public ClassC(ClassC other) {

super(other);

this.cValue = other.cValue;

}

public synchronized void displayValues() {

System.out.println("aValue = " + getAValue());

System.out.println("bValue = " + bValue);

System.out.println("cValue = " + cValue);

}

}

public class Main {

public static void main(String[] args) {

ClassC obj1 = new ClassC(10, 20, 30);

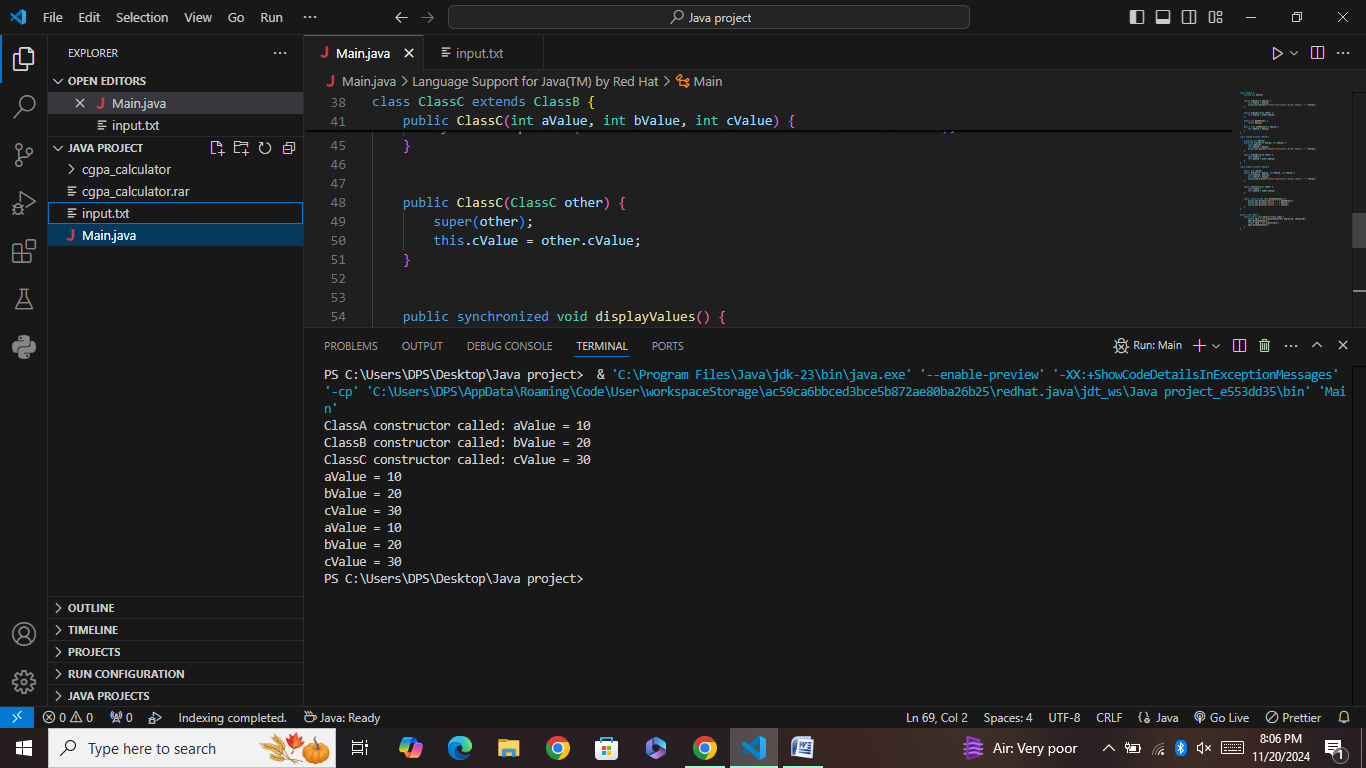
obj1.displayValues();

ClassC obj2 = new ClassC(obj1);

obj2.displayValues();

}

}



**Explanation of the Code:**

1. **Constructor Chaining**:
   * **ClassC** uses **super()** to call the constructor of **ClassB**.
   * **ClassB** uses **super()** to call the constructor of **ClassA**.
2. **Parameterized Constructors**:
   * All classes have parameterized constructors to initialize instance variables.
3. **Copy Constructors**:
   * Each class includes a copy constructor that performs a deep copy of the instance variables.
4. **Access Modifiers**:
   * Private, protected, and public access modifiers are used to ensure encapsulation. Instance variables are private, and getter methods are provided for access.
5. **Synchronized Method**:
   * **ClassC** includes a synchronized method **display()** that outputs the values of instance variables from all three classes.
6. **Object Creation Restriction**:
   * The constructor of **ClassC** is private, and a public static factory method **createInstance** is provided for creating instances of **ClassC**.